An Argument for Open-Source Software in Education – *by David D. Thornburg, Ph.D.*

Reduced technology budgets and the continuing high cost of software have conspired to prevent meaningful penetration of computers in America's schools. Thirty years ago, a few schools started bringing early personal computers into the classroom based on the idea that this new tool would allow students to learn at their own pace, and acquire new skills. In those early days, a simple computer cost about $2,500. Since that time, computer use in schools has grown, and the price of these systems dropped while the performance steadily grew. Based on early growth rates, many expected that it would only be a few years before every child had meaningful access to computers in the classroom. Furthermore, many assumed that this infusion of technology would facilitate a transformation of educational practice in ways that would benefit children in numerous powerful ways.

Boy, was that wrong! Instead of computers becoming commonplace, technology penetration bottomed out at a student/computer ratio of about 4:1, a number that has remained constant for the past five years. Instead of reaching every child, the US average technology penetration leaves 75% of our children behind. Outside the classroom, we see a different story. Children increasingly have powerful computers with broadband Internet access at home. For many (but not all) children, they have better access to the tools of our age at home than they do at school. That said, the gap in home access is between the "haves" and the "have nots" still exists. The digital divide is very real.

Public education has a special obligation to empower *every* child to learn to the best of her abilities. There is no longer any question that powerful computers coupled to the Internet can assist in this task. The question is how to afford it. With the steady decline of federal money for educational technology, and the constant pressure to upgrade commercial software, it is a miracle that we have the tools we do! But, in the final analysis, this is no excuse for failing to meet our obligation to provide the resources needed by every child in our schools.

One project that has shown what can be done to solve this problem in Indiana's inACCESS program (www.doe.state.in.us/inaccess) whose goal is to insure that every high school student in the State has meaningful access to technology. With more than 300,000 students in Indiana high schools, creative solutions were required. The approach taken in Indiana was to use powerful (but inexpensive) computers running the Linux operating system and using a lot of free open-source software (FOSS), such as StarOffice, Firefox, *etc.* This decision has the potential to reduce software costs from $100 per year *per computer* to nearly zero. To get a sense of this impact, consider a state with one million students. If you had computers for every child, each running proprietary software, you would still have to find $100 million per year just to legally turn them on.

When this project started, Linux was largely relegated to the back office as the operating system of choice for servers and other computer systems out of sight of most teachers and students. By boldly placing Linux on student desktops, Indiana decided to push the envelope at just the right time. Today's "desktop" Linux systems (*e.g.*, versions from Red Hat, Novell's SuSE, Ubuntu's Edubuntu, *etc.*) provide desktops virtually indistinguishable from those associated with computers running Windows or Macintosh operating systems. Red Hat, in conjunction with MIT's "one laptop per child" project also supports a completely new user interface, Sugar, that provides a whole new way for learners to interact with computers.

This is not to say that there aren't differences between Linux and proprietary operating systems. Linux systems typically boot up in a fraction of the time needed by a "mature" version of Windows, and can also run amazingly

well on older computer hardware, thus extending computer life in the classroom. On the downside, the popular applications are different from (but largely compatible with) their commercial counterparts. StarOffice replaces Microsoft Office, GIMP replaces Photoshop, *etc.* Because these programs are different, there is a learning curve to be overcome as the transition is made from the commercial to the FOSS version of the application, but typically this learning step isn't much different from that associated

with moving from an older version of a commercial title to a new one. In other words, if you've ever gone through an upgrade cycle, you probably have all the skills you need to master the new FOSS titles. For example, GIMP is installed automatically with SuSE Linux (SLED) and several other distributions (*e.g.*, Ubuntu), meaning that every Linux computer has a powerful graphics editor already installed for free. The provides an opportunity for teachers to let students make use of a tool whose proprietary equivalent would cost a lot of money to purchase. Opportunities for student creativity increase in the world of FOSS since there is no financial barrier to installing powerful special software on the off-chance that some students might use it. My own software mix on my Linux laptop would cost (at educational discounts) more than $500 to replace with commercial titles. Again, the cost savings per machine is significant. Add to this the fact that many of these titles are cross-platform, and can be given to students to take home, and the benefits of FOSS grow even larger.

Going back to Linux, consider the current push by Microsoft to switch users to Vista. To run properly, this operating system requires more computer power than is commonly found on student desktops. This means that current Windows XP users will be stretching the life of a now-obsolete operating system that Microsoft can choose to stop supporting whenever they wish. And, even if schools decide to equip all new computers with Vista, they will still be using XP on older systems, meaning they will be supporting two operating systems. In this case, the benefit of adding Linux to the mix is increased. If you are supporting two operating systems, and one of them lets you extend the life of existing computers, the benefits to the school's budgets are tangible.

Indiana's push to Linux and FOSS may have been driven by cost, but that is not the only factor. While cost is important, it can't be the deciding factor: quality is essential. If a free alternative is not as good, or better, than a commercial product, then quality must win out over price. We must never treat schoolchildren as second-class citizens. We who care about education must always put children first. Fortunately (as will be illustrated later), the quality of many FOSS titles is amazing. Some of the titles of greatest value in K-12 education have features not found in *any* commercial titles – features that are of great value to students.

If all I've said is true, why haven't schools throughout the nation raced to embrace this approach? There are several reasons, usually expressed as concerns about open-source and Linux. I think these questions come from the fact that FOSS represents a new paradigm. If we start showing educators powerful ways to use technology that save tremendous amounts of money, their natural instinct will be to look for flaws. After all, if we can now do great things for free, this might imply that we've been spending scarce resources inappropriately in the past. To be fair (and to put this idea to rest), it has only been within the past few years that Linux could be considered as a desktop operating system for anyone except strong technology enthusiasts. But today we are truly in a new world. To bring this world to fruition, we need to honestly address people's concerns. Unless we do this, the true 1:1 computer revolution in education will continue to be stalled. In the following paragraphs I attempt to honestly address some of the common questions that are raised when the topic of FOSS and Linux is brought up among educators.

I've spent a tremendous amount of time and effort to become certified in other operating systems, why should I now go through this process for Linux?

Certification is an ongoing process. Every time a new version of an old operating system hits the streets, technology directors have to get up to speed. Fortunately, some of the larger Linux providers (*e.g.*, Novell) understand this need and are addressing it. The reality is that, once Linux gets into the classroom, most students scarcely know the difference. Depending on how the desktop is configured, applications launch and run the same as they always have. The only difference that might be noticed is that older computers now "run faster," and there is no more "blue screen of death" with which to contend.

If software is free, how can it be any good?

Most FOSS is written by people who intend to use it themselves. For this reason, they want it to be as good as possible. Some developers created programs to address missing features in existing commercial titles – features the commercial developer had no intention of adding. Also, most popular FOSS titles are maintained by informal teams scattered all over the world. Bugs get identified and fixed quickly as a result.

If no one is getting paid, how is FOSS and Linux maintained?

First, many software developers are getting paid for their efforts. Some corporations who rely heavily on certain FOSS titles provide time for their own employees to maintain the software. This has direct benefit to the company, as well as the global community of users. Also, the maintainers of the software have their names included in the source code, and many of them communicate with each other on a regular basis. A lot of FOSS is developed at Universities and government agencies (*e.g.*, NASA), and these developers are being paid for their efforts.

Who do you call when you need help?

There are two answers to this question. First, commercial versions of Linux (*e.g.*, Novell's SLED) charge an annual service fee that makes sure you have the latest upgrades, and access to a special help desk where you can post questions, report bugs, *etc.* My experience has been that many issues get resolved within a day of being reported. Alternatively, specific FOSS titles generally have e-mail addresses for the main developers who are eager to hear from users about any problems they might be having, as well as handling requests for new features. Again, my experience is that bug fixes take about a day, unless they involve a hardware conflict.

But if you ask this question about FOSS, you need to ask it about commercial software as well. I can't think of a single piece of commercial software for which I had a bug fix in a day. Typically, I receive an e-mail stating "this is a known problem that will be addressed in the next release."

What if a critical application gets discontinued?

This is a powerful question given that many FOSS titles are created because a small number of people wanted to have the program. What if these folks decide to do something else with their lives? The answer lies in the "open-source" phrase. This means that the raw source code for the software is open to anyone who wants it. You can (if you wish) download the raw source code for any piece of FOSS. You can add features, remove others, make your own upgrades, or do anything you wish as long as you have the requisite programming skills. Because FOSS is generally a group effort, the source code is generally well documented. This way, if a product is discontinued, you can maintain it yourself as long as you wish.

Contrast this with commercial titles that go out of print. There are some great educational titles from the 1980's that are no longer available. Because the source code for these programs is proprietary, there is no legal way for you to upgrade or use these programs once they have been taken off the market.

Why should I use Linux if it doesn't run the applications I need?

There are some commercial titles that are so compelling they deserve to have their own computers dedicated to them. These titles may only run on one platform – Windows, for example. In that case it might appear that Linux users are out of luck. Furthermore, if these programs are "mission critical" applications, then it would seem that there is no choice but to stay with a proprietary operating system.

Fortunately, there is a commercial solution to this problem called *Crossover Office* (www.codeweavers.com) that allows many Windows applications to run under Linux. Because the cost of this software is significantly less than the cost of Windows XP (for example), this solution can solve the problem.

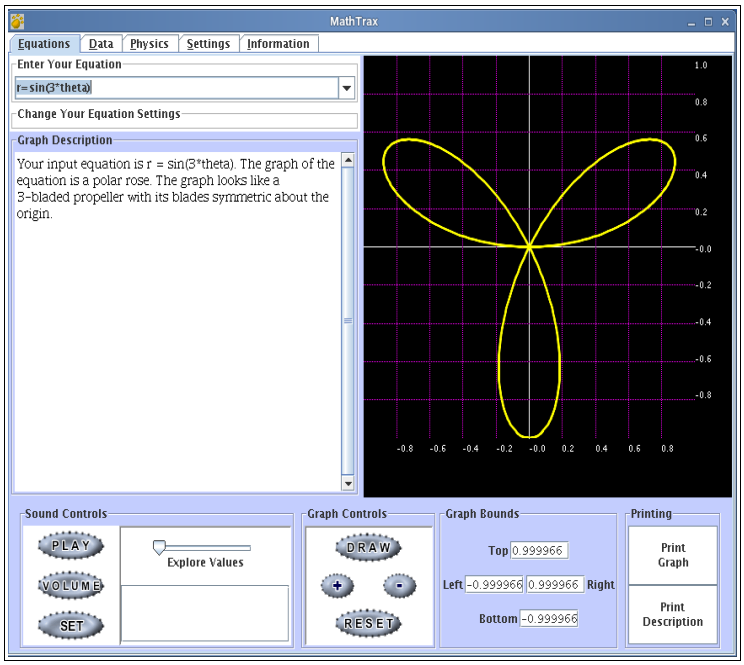
In cases where this approach doesn't work, schools need to adjust themselves to having several operating systems available. For example, every student computer could be running Linux, and a few specialized computers running Windows or Mac OS X (for high-end video production, perhaps) could be in a special lab for those projects that need the special software. This approach has several benefits. First the specialized computers would only be running one or two programs, thus keeping them "clean" from the mess that can come from running numerous programs on a single (non-Linux) computer.

Earlier in this brief, I suggested that, while cost is undeniably a factor, the driving force must always be quality. I have argued that, on this basis, FOSS running on Linux can be a winning combination. The following example describes a piece of math software that runs on all platforms (Linux, Windows, Macintosh). This program is typical of the kind of quality applications that are available for free if you are willing to look for them. Rather than provide a laundry list of great titles, I chose to look at just one. But rest assured that there are many, many more titles I could have chosen just as well.

MathTrax

There are numerous examples of FOSS that perform as well, or better, than their commercial counterparts. To make the point that such products exist I will examine one program written under the support of NASA: MathTrax (http://prime.jsc.nasa.gov/mathtrax/).

At its core, MathTrax is a program for plotting and exploring mathematical functions. The motivation behind this software was meeting the needs of the visually impaired learner, but the features it has makes it perfect for all students.



This figure shows a polar plot of the function $r = \sin(3\theta)$. I just typed in the equation, and MathTrax did the rest. In addition to plotting the graph, MathTrax also created a text description of the function. To my knowledge, this feature of MathTrax is unique. Using the Math Description Engine developed by NASA, the equation was analyzed and a plain text description of the graph was generated automatically. This feature is simply astounding. Furthermore, NASA makes the Math Description Engine available to anyone who wants to incorporate it into their non-commercial software.

And, if that wasn't enough, the "Play" button at the lower left of the screen plays a stereophonic musical tone that changes as a cursor moves around the graph, letting visually impaired students "hear" what various graphs sound like.

To be honest, I can't imagine *any* student who wouldn't benefit from this program. By representing mathematical functions through graphs, text, and sounds, students have multiple pathways to understanding not found elsewhere.

MathTrax is just one example of FOSS that should be used by students throughout the country because if its quality, not just because it is free. I could just as easily explored other titles. They are easy to find. My main point is that FOSS and Linux would make sense even if you had to pay for them.

In fact, I think a good litmus test for any software or operating system is simply this: Would I use this tool if I had to pay for it myself? If the answer is "yes," then you have a great piece of software. If it is "no," then the software is no bargain, even if it is free.

Increasingly, Linux and a whole host of FOSS titles are making it easier to say "yes," and this means we are well on the way to solving the access problem confronting most schools in the United States today.

Indiana is paving the way in the United States, and there is little question that the inACCESS project is a driving force for bringing meaningful access to computers to the hands of every student, for the first time in history.